REMARKS

I. INTRODUCTION

Claims 1-7 are pending in the present application. No new matter has been added. In view of the following remarks, it is respectfully submitted that all of the presently pending claims are allowable.

II. THE 35 U.S.C. § 103(a) REJECTIONS SHOULD BE WITHDRAWN

The Examiner has rejected claims 1-7 under 35 U.S.C. § 103(a) as unpatentable over U.S. Patent No. 6,529,985 to Deianov et al. (hereinafter "Deianov") in view of U.S. Patent No. 6,658,571 to O'Brien et al. (hereinafter "O'Brien"). (*3/18/05 Office Action*, ¶ 3).

Deianov describes a system for selective interception of system calls using system call wrappers to execute in a process address space of computer memory. (*Deianov*, col. 3, lines 30-34). Pointers to system calls that are to be intercepted and which are normally stored in an interrupt vector table are replaced with pointers to an interception module. (*Id.*, col. 3, lines 36-38). The interception module maintains an association table of the select processes and system call wrapper entry points. (*Id.*, col. 3, lines 58-62). The module determines from the association table whether the calling process is one of the selected processes with a registered entry point in a system call wrapper. If so, the interception module prepares to call the appropriate system call wrapper. (*Id.*, col 4, lines 5-13).

O'Brien discloses a security framework for wrapping software applications and restricting access to system resources. (*O'Brien*, Abstract). O'Brien uses a hyperviser, which is a layer of software disposed between hardware and an operating system that implements the same

PAGE 5/10 * RCVD AT 5/18/2005 2:36:20 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/1 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):03-56

instruction set as the hardware, within the the kernel but between the operating system and other software applications (*Id.*, col.2 line 62 - col. 3, line 1). The use of the hyperviser allows the computing system to "control access to computing resources . . . such as memory, files, network sockets and processes." (*Id.*, col. 3, lines 2-9). More specifically, the security framework utilizes security modules which grant or deny access to computing resources based on either the application requesting access or the computing resource being requested. (*Id.*, col. 3, lines 38-45). Thus, each security module wraps the applications thereby preventing the applications' access to the resources unless authorized by the modules.

Claim 1 recites a method including "*determining a current processing mode of an executing software function*" and "when the current processing mode is a privileged processing mode, executing a direct program flow control instruction to directly access an instruction within software having the privileged processing mode" and "*when the current processing mode is an unprivileged processing mode, executing an indirect program flow control instruction to cause execution of the instruction within software having the privileged processing mode.*" The Examiner has correctly recognized that Deianov does not teach or suggest "determining a current processing mode of an executing software function" as recited in claim 1. (*3/18/05 Office Action*, ¶¶ 4-5). To cure the deficiency of Deianov, the Examiner has cited O'Brien. (*Id.*, ¶ 6).

It is respectfully submitted that O'Brien neither discloses nor suggests "determining a current processing mode of an executing software function" and "when the current processing mode is an unprivileged processing mode, executing an indirect program flow control instruction to cause execution of the instruction within software having the privileged processing mode," as recited in claim 1. In O'Brien, the security module enforces:

PAGE 6/10 * RCVD AT 5/18/2005 2:36:20 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/1 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):03-56

> [A] set of rules identifying which computing resources 106 the browser is allowed to access as well as what permissions the browser has. If there is no rule that allows access to a computing resource 106, then the security framework 101 refuses any requests to access that computing resource 106.

O'Brien, col. 7, lines 27-33. Reading preset rules or permissions is not a determination of a current processing mode. In the present invention, the examination of the "current processing mode" is used to decide whether an executing software function should be allowed to access a system resource directly or indirectly and not whether it is allowed to access the resource or not. The software functions of the present invention are presumed to have access to system resources. Thus, the present invention allows the computing system to decide the type of access that the software functions will obtain – direct or indirect.

In further support of the rejection, the Examiner states that Deianov teaches determining whether the executing software function can access the system resource directly or indirectly, and, O'Brien, therefore, is being cited solely to show the step of determining the access level of the executing software function to a system resource. (*3/18/05 Office Action*, ¶¶ 4, 17). Applicant respectfully disagrees with the Examiner's reading of Deianov and O'Brien, and the combination of their disclosures.

Initially, it is respectfully submitted that Deianov does not disclose or suggest determining whether the executing software function is to be intercepted. Deianov only refers to system calls which are to be intercepted, and those which are not intercepted. In contrast to the Examiner's assertion, it is respectfully submitted that at no point does Deianov disclose how system calls are distinguished between intercepted and not-intercepted. In fact, Deianov describes the method of a process executing a system call as follows:

PAGE 7/10 * RCVD AT 5/18/2005 2:36:20 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/1 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):03-56

> Executing processes make system calls 115. When a process makes a system call 115 *that is to be intercepted*, the interception module 111 executes. The interception module 111 examines the association table 127 to determine whether the process that the made the system call 115 is associated with a system call wrapper 125.

(*Deianov*, col. 8, lines 14-19) (emphasis supplied). Thus, only system calls which are intercepted are examined by the interception module. Deianov does not disclose or suggest how to distinguish between system calls which are to be intercepted and those that are not intercepted, or what happens to the system calls that are not intercepted.

Further, one of skill in the art would not be motivated to combine the teachings of Deianov with those of O'Brien. Specifically, as noted above, Deianov intercepts system calls and then determines whether the system call should utilize a pointer or execute a system call wrapper. (*Deianov*, col. 8, lines 14-19). In contrast, O'Brien teaches that each system call is turned to a security module which initiates pre-call processing. If the pre-call processing fails, the security module sends a result indicating an error to the application which made the system call. (*O'Brien*, col. 5, lines 56 - col. 6, line 16). Thus, the security module in O'Brien blocks certain system calls from the system resources. Therefore, while Deianov provides access to the system resources for intercepted system calls through different channels, O'Brien restricts access to the system resources for certain system calls.

It is respectfully submitted that neither Deianov nor O'Brien, either alone or in combination, disclose or suggest determining a current processing mode of an executing software function" and "when the current processing mode is an unprivileged processing mode, executing an indirect program flow control instruction to cause execution of the instruction within software having the privileged processing mode," as recited in claim 1. Because claims 2-5 depend from,

PAGE 8/10 * RCVD AT 5/18/2005 2:36:20 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/1 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):03-56

and, therefore include all of the limitations of claim 1, it is respectfully submitted that these claims are also allowable.

Claim 6 includes substantially the same limitations as claim 1, including "determining a current processing mode of the program code segment" and "execute an indirect program flow control instruction if the current processing mode is an unprivileged processing mode." Thus, for the reasons described above with reference to claim 1, it is respectfully submitted that claim 6 is also allowable.

Claim 7 includes substantially the same limitations as claim 1, including "determining a current processing mode of the program code segment" and "execute an indirect program flow control instruction if the current processing mode is an unprivileged processing mode." Thus, for the reasons described above with reference to claim 1, it is respectfully submitted that claim 7 is also allowable.

PAGE 9/10 * RCVD AT 5/18/2005 2:36:20 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/1 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):03-56

CONCLUSION

In light of the foregoing, Applicant respectfully submits that all of the now pending claims are in condition for allowance. All issues raised by the Examiner having been addressed, an early and favorable action on the merits is earnestly solicited.

Respectfully submitted,

Dated: May 18, 2005

By: _____
Michael J. Marcin (Reg. No. 48,198)

Fay Kaplun & Marcin, LLP
150 Broadway, Suite 702
New York, NY 10038
(212) 619-6000 (tel)
(212) 619-0276 (fax)

PAGE 10/10 * RCVD AT 5/18/2005 2:36:20 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/1 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):03-56